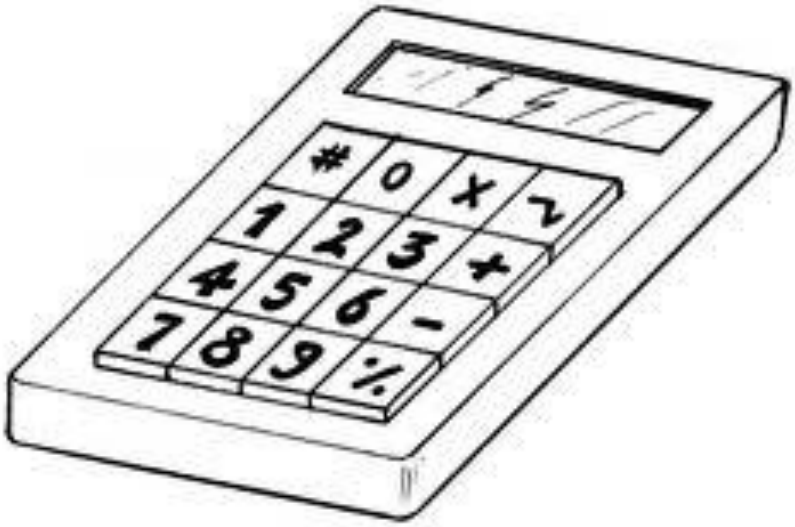
Ingénierie des réseaux

Projet Socket-calculatrice

Yorly ORELLANOS
Mathieu CASTETS
Thibault ETIENNE

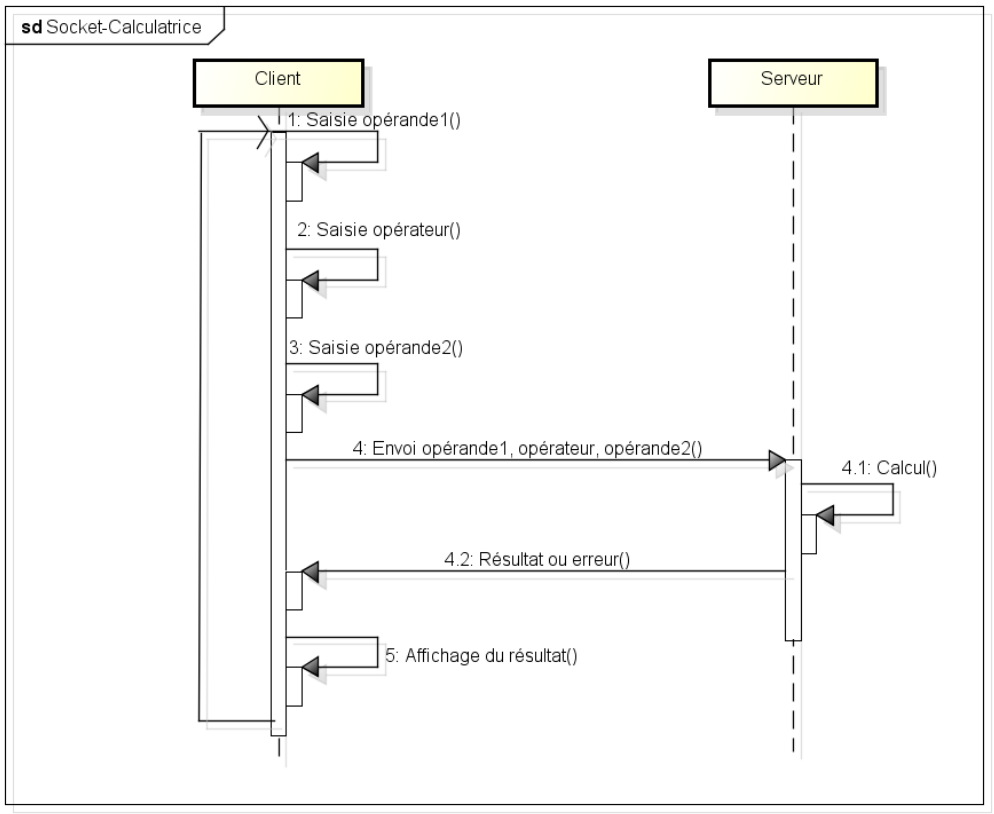


Conception

Pour réaliser cette application, nous avons commencé par imaginer le comportement désiré. Ceci est résumé dans le tableau ci-dessous :

Qui	Action	Message
Le client	Propose à l'utilisateur de saisir un nombre	Opérande 1 :
Le client	Propose à l'utilisateur de saisir un opérateur arithmétique	Opérateur : +, -, *, /
Le client	Propose à l'utilisateur de saisir un second nombre	Opérande 2 :
Le client	Envoi les informations au serveur et attend sa réponse	J'ai envoyé au serveur l'opération …
Le serveur	Reçoit les informations données	*Pas de message*
Le serveur	Fait le calcul	*Pas de message*
Le serveur	Envoie le résultat de l'opération réalisée	*Pas de message*
Le client	Affiche le résultat	Résultat : …
Le client	Propose de faire un nouveau calcul	Autre opération ? (O/N)

Nous avons modélisé ce fonctionnement dans le diagramme suivant :



Nous avons décidé que les communications par le biais des sockets se feraient via le port 12345 afin de ne pas interférer avec les ports communément utilisés. En effet, ces ports sont réservés à des applications spécifiques et sont déjà occupés par d'autres interlocuteurs, ce qui provoque un conflit. Exemple : lors d'un essai de l'application sur le port 80, la réponse est « HTTP/1.1 400 Bad Request ».

Pour l'échange des données, le protocole TCP est utilisé car il permet un contrôle des données et assure ainsi la fiabilité des données en transit pour l'application.

Réalisation

Utilisation des décorateurs

Lors de l'utilisation des entrées (saisie des données) et sorties (affichage), nous avons utilisé des décorateurs car ils fournissent une interface adaptée aux traitements des données. On notera notamment l'utilisation du *BufferedReader* qui utilise un buffer de lecture ou de *OutputStreamWriter* qui transforme des flux de caractères en flux d'octets.

Gestion des exceptions

Nous avons mis en place un système de gestion des exceptions afin d'afficher des messages d'erreurs facilement compréhensibles par l'utilisateur. Ceci évite aussi une fermeture inopinée de l'application. Cette gestion des exceptions est d'autant plus importante lors du calcul (conversion arithmétique, division par zéro, …).

Difficultés rencontrées

Nous avons été confrontés à plusieurs problèmes lors de la réalisation de ce projet.

Le premier a été celui de la conversion des données saisies par l'utilisateur pour leur traitement par le serveur. En effet, nous avons souhaité travailler avec des nombres de type *float* et avons donc dû les convertir en chaines de caractères afin de pouvoir les faire transiter à travers les sockets.

Dans un second temps, une fois la calculatrice fonctionnelle pour effectuer une unique opération, il nous a fallu modifier l'application afin que le client puisse faire plusieurs opérations sans relancer le logiciel. Enfin, nous avons souhaité que le serveur reste actif lorsque le client est fermé par l'utilisateur après avoir effectué au moins une opération. Lors de l'implémentation de ces fonctionnalités, nous avons rencontré un problème dû à une mauvaise utilisation des sockets de notre part. En effet, nous n'ouvrions qu'une socket pour l'ensemble des calculs alors qu'il est nécessaire d'en créer une à chaque calcul demandé par l'utilisateur.

Limitations du logiciel

Cette calculatrice est limitée à faire les opérations arithmétiques de base que sont l'addition, la soustraction, la multiplication et la division.

Manuel d'utilisation

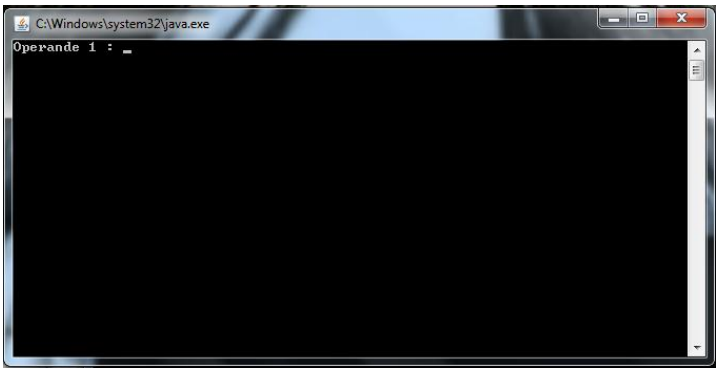
Lancement du logiciel

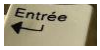
Pour lancer la calculatrice, il faut d'abord ouvrir le fichier exécutable « 1 - Serveur.bat » puis lancer le fichier exécutable « 2 - Client.bat ». Cela permet à l'utilisateur d'éviter de devoir lancer des commandes dans un invité de commande. Le numéro de port et l'IP sont aussi spécifiés dans les .bat. Il est donc aisé de les modifier en ouvrant ces fichiers avec un éditeur de texte quelconque.

Cette calculatrice comprend un certain nombre de fonctions de base, dont les fonctions addition, soustraction, multiplication, division.

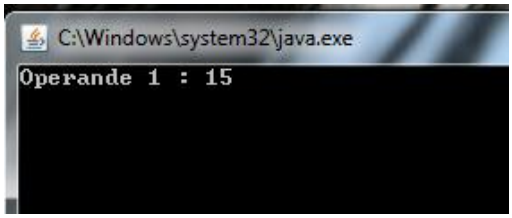
Calcul

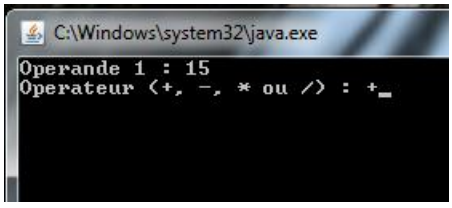
Avec les calculs manuels, vous entrez vos formules de gauche à droite, simplement comme elles s'écrivent sur une feuille de papier. Avec les operateurs qui comprennent +, -, *, /, la calculatrice applique automatiquement la logique du système pour calculer le résultat.



1. Entrez le premier chiffre à calculer, appuyez sur la touche Entrée 

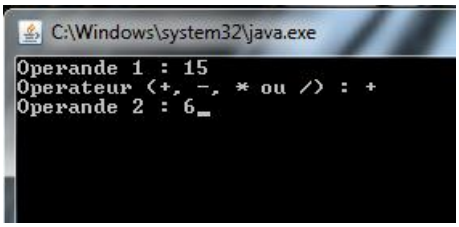
pour valider, puis vous devez sélectionner l'operateur arithmétique +, -, * et /.





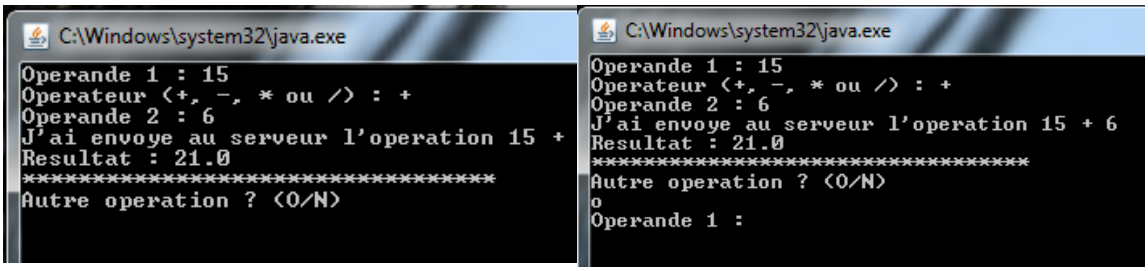
2. Entrez le deuxième chiffre à calculer et appuyez sur la touche Entrée

pour afficher le calcul.



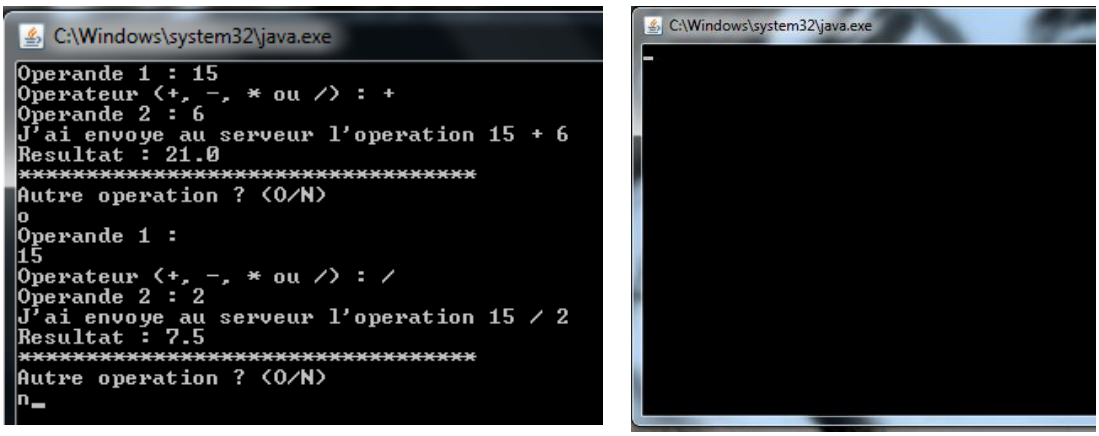
3. Le client envoi l'information au serveur, celui-ci valide que l'information transmit par le client est correct, suite le serveur transmet la réponse et demande si l'utilisateur veut faire une autre opération arithmétique.

   Appuyez sur O pour exécuter le nouveau calcul ou N pour quitter la calculatrice.



   En cas l'utilisateur appui sur la touche O, il pourrait refaire tout le processus

4. Si l'utilisateur appuis sur la touche N, le client se déconnecté et le serveur reste connecté en attend d'une nouvelle transmission.

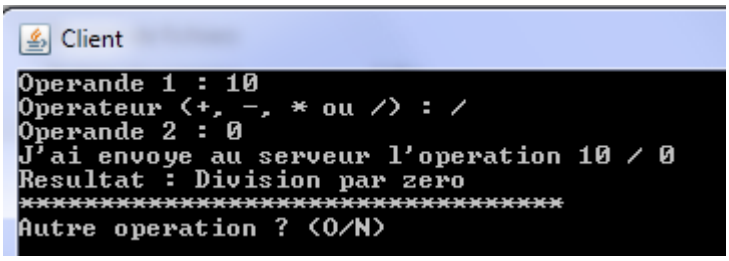


Erreurs

Nous avons géré plusieurs cas d'erreurs qui peuvent survenir pendant la réalisation d'une opération. Ainsi, le serveur renvoi des messages formatés correspondant au type d'erreur levée. Voici la liste des erreurs possibles :

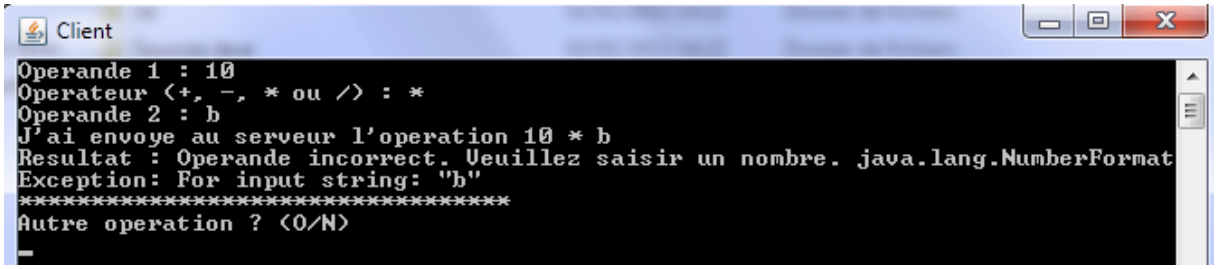
- Division par zéro :

Cette erreur survient lorsque l'opérande choisit est celui de la division '/' et que le second opérande a pour valeur 0 ;


Client
Operande 1 : 10
Operateur (+, -, * ou /) : /
Operande 2 : 0
J'ai envoye au serveur l'operation 10 / 0
Resultat : Division par zero
****
Autre operation ? (O/N)

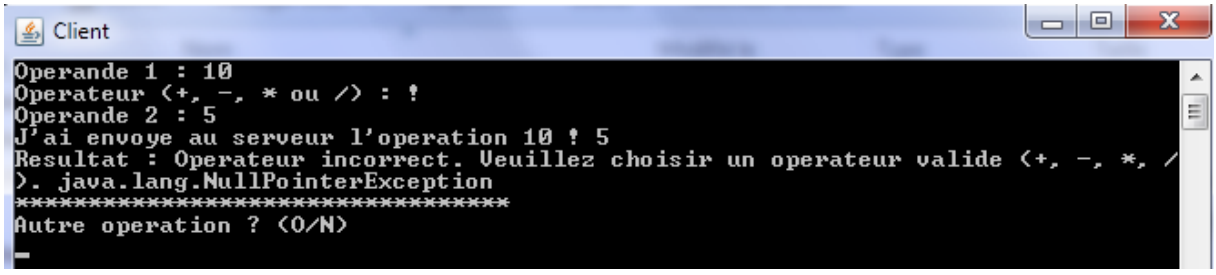
- Opérande incorrect :

Cette erreur survient lorsque l'utilisateur saisit un opérande qui n'est pas un nombre ;


Client
Operande 1 : 10
Operateur (+, -, * ou /) : *
Operande 2 : b
J'ai envoye au serveur l'operation 10 * b
Resultat : Operande incorrect. Veuillez saisir un nombre. java.lang.NumberFormat
Exception: For input string: "b"
****
Autre operation ? (O/N)

- Opérateur incorrect :

Cette erreur survient lorsque l'utilisateur ne saisit pas un des 4 opérateurs définis que sont l'addition '+', la soustraction '-', la multiplication '*' et la division '/' ;


Client
Operande 1 : 10
Operateur (+, -, * ou /) : !
Operande 2 : 5
J'ai envoye au serveur l'operation 10 ! 5
Resultat : Operateur incorrect. Veuillez choisir un operateur valide (+, -, *, /
). java.lang.NullPointerException
****
Autre operation ? (O/N)

- Fermeture prématurée du client :

Cette erreur survient lorsque le client est soudainement fermé alors qu'il n'a envoyé aucune donnée au serveur.